

Optimizing Data Access With Visual Basic® 6.0 Part II

**William R. Vaughn
Developer Trainer
Microsoft Technical
Education**

A large space shuttle is shown launching vertically on the right side of the image. It has a white body with orange and black stripes. Bright orange and yellow flames and white smoke are coming from the engines at the bottom. In the top right corner, there are small icons of a calendar and a document connected by lines.

POWER

Windows DNA 2000

Readiness Conference

/// featuring SQL Server 2000

William R. Vaughn

Developer Trainer, Microsoft Corporation

Author:

Hitchhiker's Guide to Visual Basic & SQL Server (now in 6 languages)

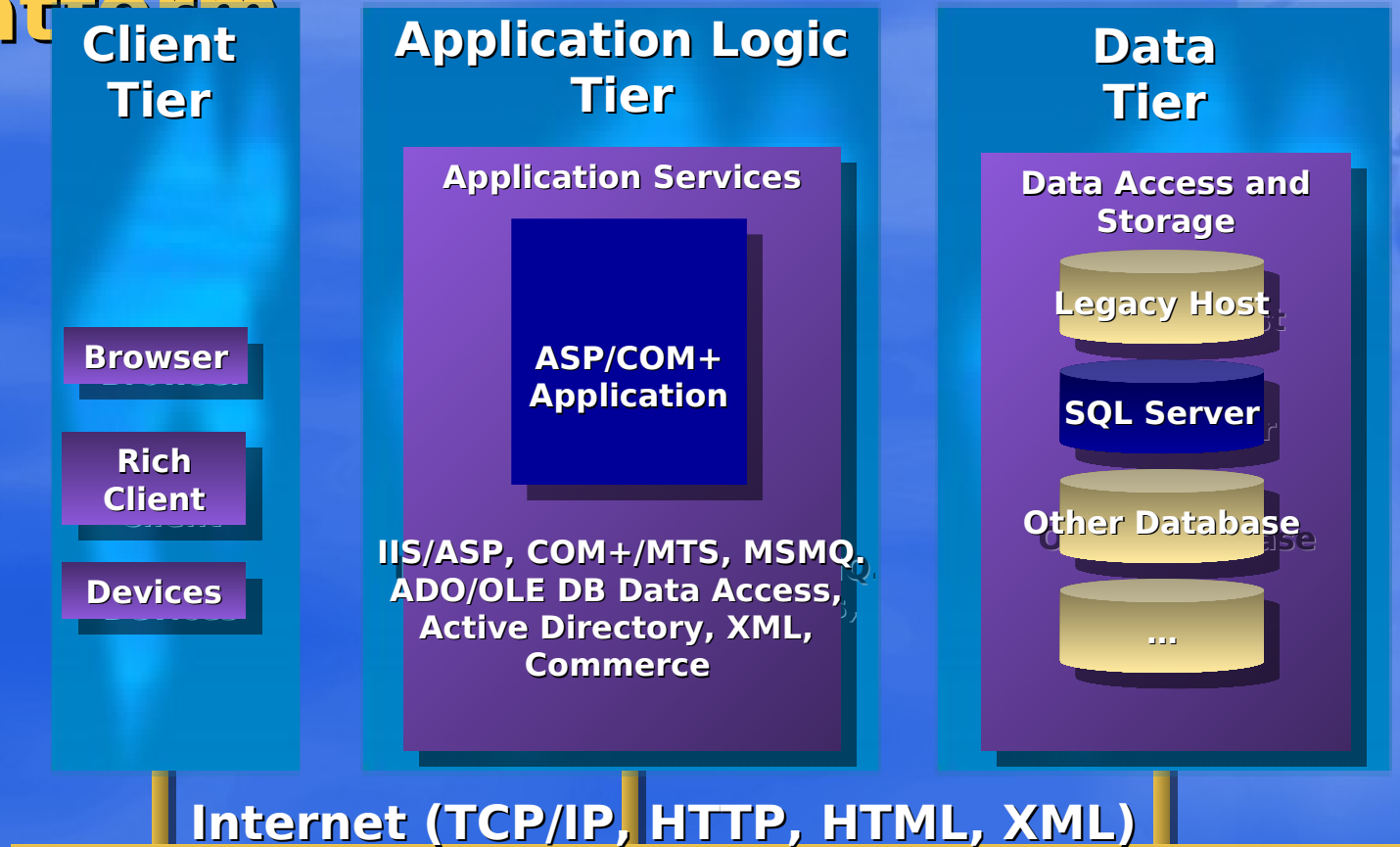
billva@nwlk.com

www.betav.com



Windows DNA 2000

Next Generation Web Application Platform



Microsoft
SQL Server 2000
Server2000
Microsoft
Application Center 2000



Microsoft
Commerce Server 2000
Microsoft
Host Integration Server 2000

Microsoft
BizTalk Server 2000

Optimizing Data Access

Part II

Using ADO 2.5 with Visual Basic 6.0

- Passing data between tiers
- Writing smarter ADO code

Microsoft

**Universal Data
Access**



Introduction To ADO

2.5

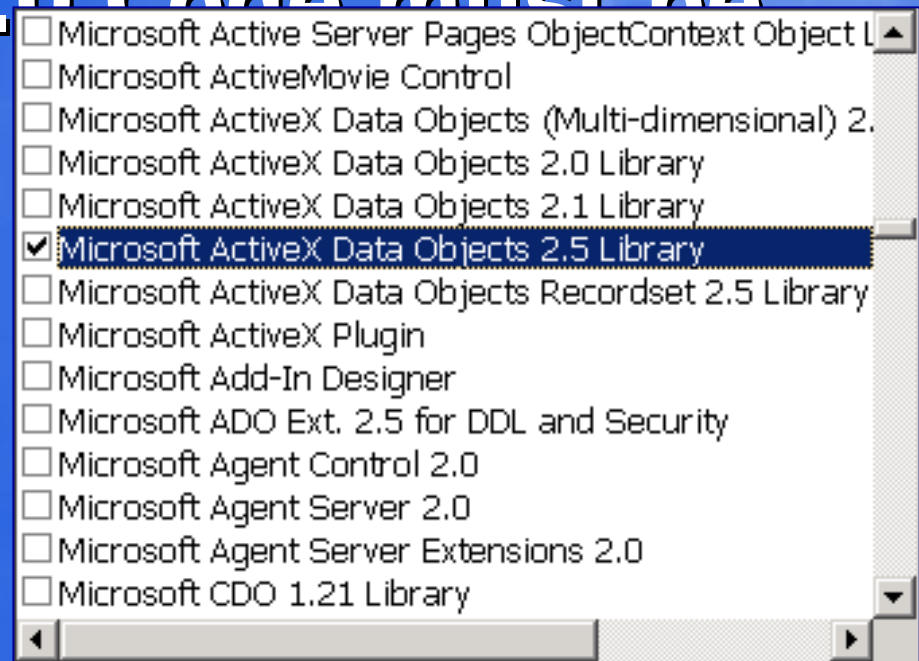
- **ADO - a COM (object) interface to OLE DB**
- **Installed from Web:
<http://www.microsoft.com/data/download.htm>**

ADO 2.5 In Visual Basic 6.0

- Reference ADO 2.5 (MSADO15.DLL)

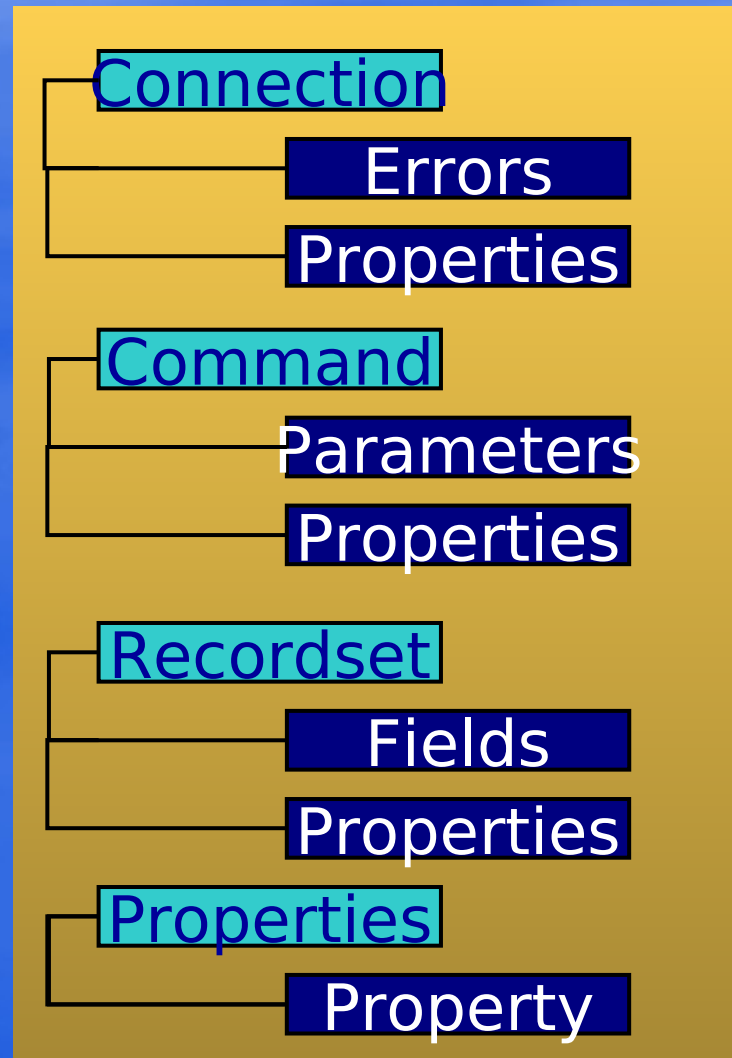
- Visual Basic 6.0 Code must be adapted to Environment Designer events

- ADO Data control
- Data Form Wizard
- Visual Data Manager (source available)



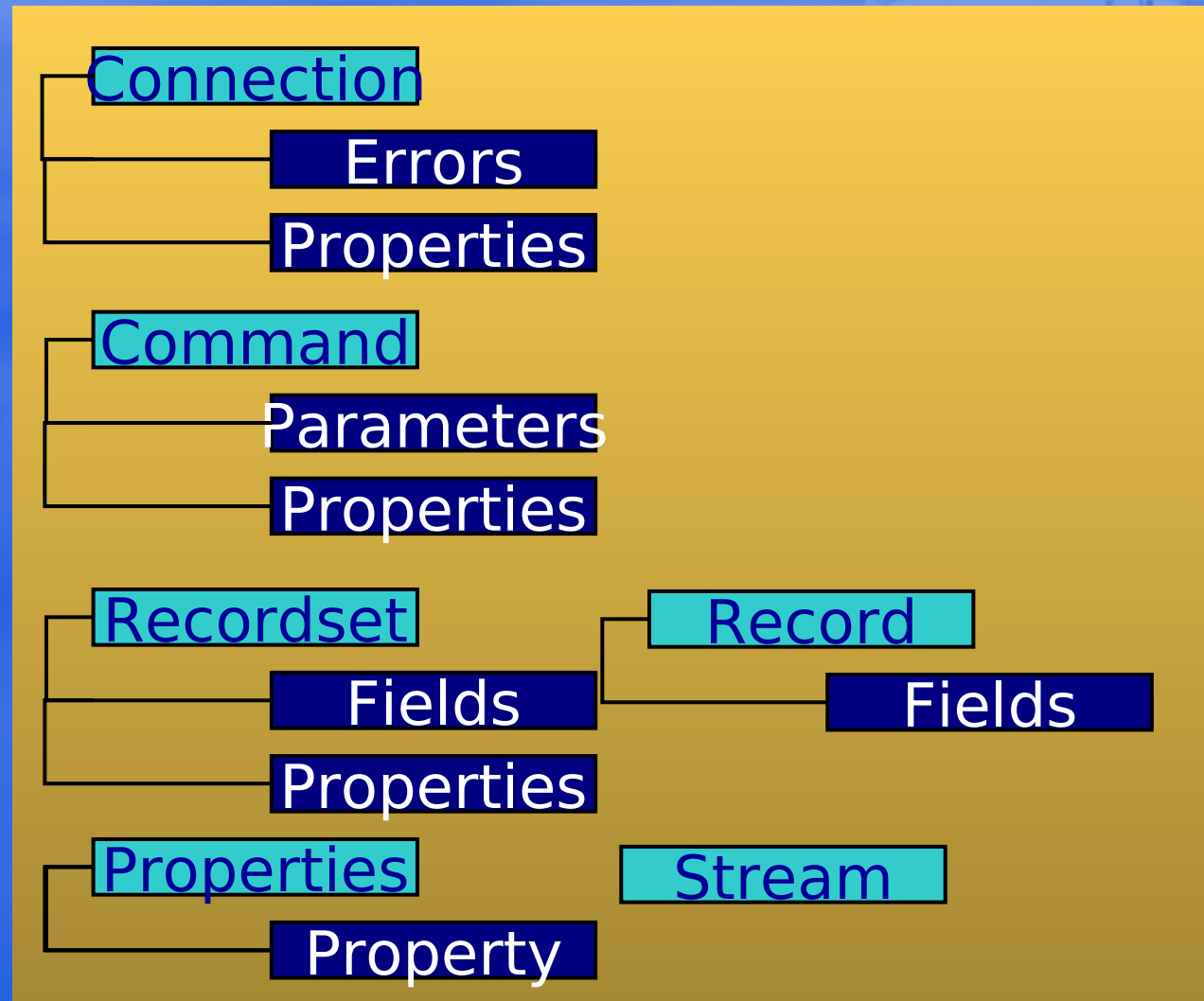
The ADO 2.0 Object Model

**ADO
2.0**



The ADO 2.5 Object Model

**ADO
2.5**



ADO 2.5 - New objects

- **Record object** - for directories, files, folders ...
- **Stream object** - manages a binary byte stream
- **To pass persisted Recordsets between layers**
 - **Recordset → XML → Stream → XML → Recordset**
- **URL Usage** - alternative to named data store
 - **Instead of connection strings, command text**

ADO 2.5...

- **New error handlers**
 - **Fewer “Error occurred” messages**
 - **Same number, different .Description**
- **XML persists hierarchical Recordsets**
- **Jet OLE DB provider for Jet 4.0 databases**
 - **Supports Table-Index paradigm**

ADO 2.5

Implementation

- **Passing data from tier-to-tier**
 - **Recordset objects**
 - **Streams**
 - **XML**
 - **Variant arrays**
 - **Delimited strings**
 - **User-defined data structures**
 - **PropertyBag objects**

ADO 2.5

Implementation

- Passing Recordsets from tier-to-tier

GetStock Quotes by Symbol

File

MSFT

Current Price \$111.4375

Change \$1.4375

Get Price

Interval 1

Recordset

DLL Method call

Stream object

```
Public Function Retrieve(Symbol As String) As Stream
    If Len(Symbol) > 8 Or Len(Symbol) < 2 Then
        Err.Raise 1, "clsGetStock", "Symbol passed is too long"
    Else
        URL = WebHost & Symbol
        Set xml = New MSXML.XMLHTTPRequest
        xml.Open "POST", URL, False
        xml.send Symbol
        strResp = xml.responseText

        sngQuote = FindText(Symbol, strResp, strLastSale,
        sngNetChange = FindText(Symbol, strResp, strChar
        intFont = FindText(Symbol, strResp, strChange, str
        Set Retrieve = MakeStream(sngQuote, sngNetChan
    End If
End Function
```

ADO Data Conversions

Recordset

Recordset

File



File



XML
Stream



Recordset



Title	r_Published	ISBN	PubID
An Introduct	1995	0-0230362-0	715
Appl...	1998	0-0230362-1	19
Business Pr	1993	0-0233058-5	156

Variant Array



Database

OUTPUT
Parms

Command

Stream



Stream



Delimited String

PropertyBag

Offloading And Persisting

- **GetRows** → variant array
- **GetString** → delimited string
- **Save method:** persists to...
 - **ADTG** — ADO 1.0 on
 - **XML** — ADO 2.0
 - **Hierarchical XML** — ADO 2.5
 - **Stream** — ADO 2.5
- **Update** → to data provider
- **Execute method**
 - **INSERT, UPDATE SQL operations**
 - **Command object parameters**

Extracting And Converting

- **Open method:** accepts data from
 - **Data source**
 - **ADTG - ADO 1.0 on**
 - **XML - ADO 2.1**
 - **Hierarchical XML - ADO 2.5**
 - **Stream - ADO 2.5**
 - **URL - ADO 2.5**
- **Command object**
 - **OUTPUT, return status parameters**

Passing ADO Data Between Tiers

- **Consider memory expenses**
 - Cache on server, client
 - Cursor location
- **Consider LAN expenses**
 - Unicode or ANSI
 - Padded structures or raw binary
- **Consider session/server state**
 - Loss of scalability
 - Slow user/system makes matters worse

Passing ADO Data Between Tiers

- **Consider client app's intelligence**
 - **Ability to process 2.5 structures?**
 - **Code to bind, update, decipher, display structure**
 - **ADO capable?**
- **Consider developer's skill**
- **Limit size of rowset, locks, overhead**

Recordset Services

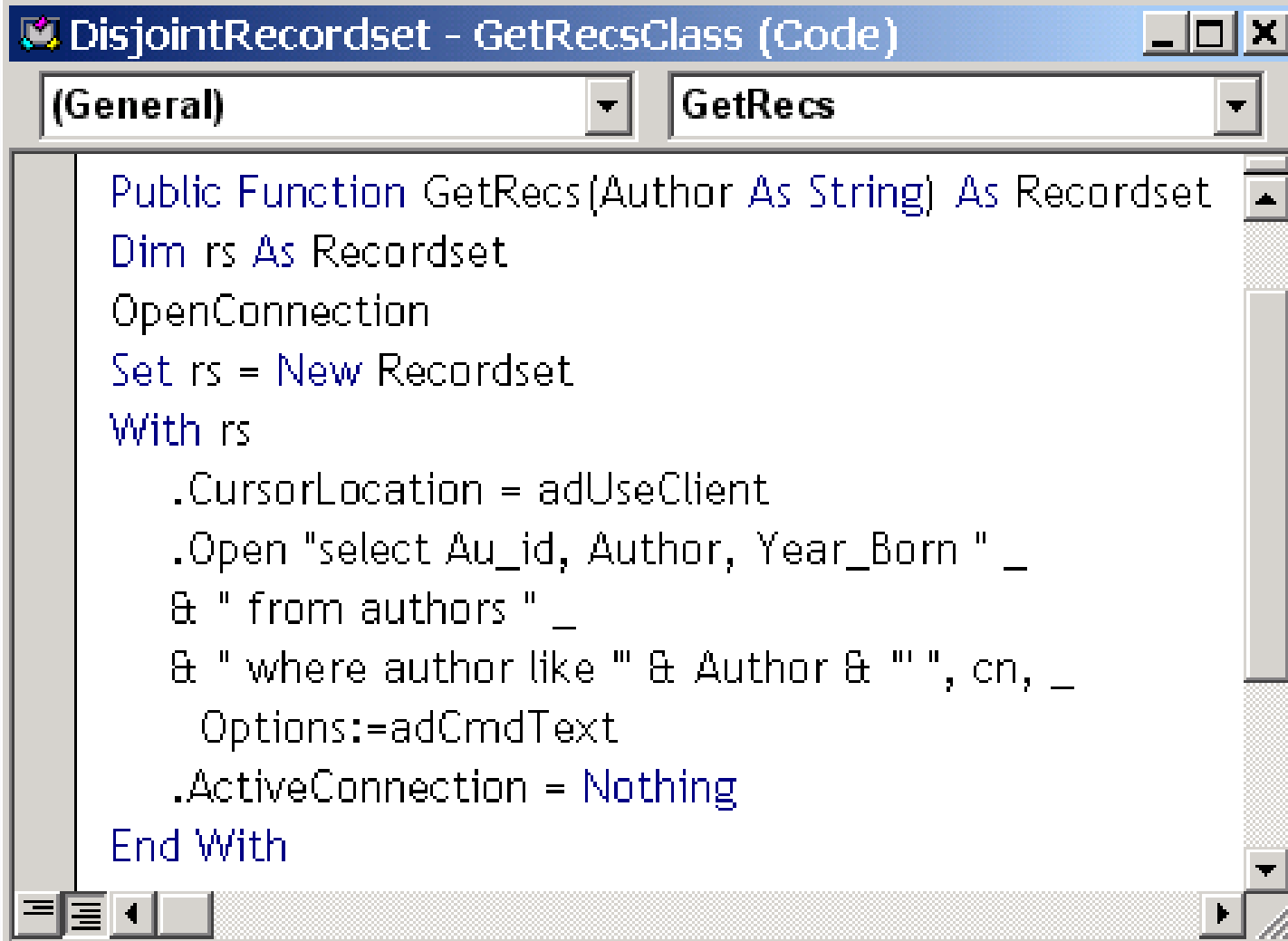
- **Manages (multiple) rowset/result set(s)**
- **Cursor services**
 - **Scrolling, binding, updatability**
- **Source/persistence from/to**
 - **Files, streams, Recordsets, providers**
- **Support by controls**
 - **Recordset, DataSource property**
- **Full data description language (DDI)**

Passing Disjoint Recordsets

- Use Client(batch)
CursorLocation
- Open Recordset as usual
- Assign (don't Set)
ActiveConnection = Nothing
- Can be updatable, or RO
- Use RecordCount for row
count
(when available)
- Includes record status

Passing Recordsets...

Server-side code



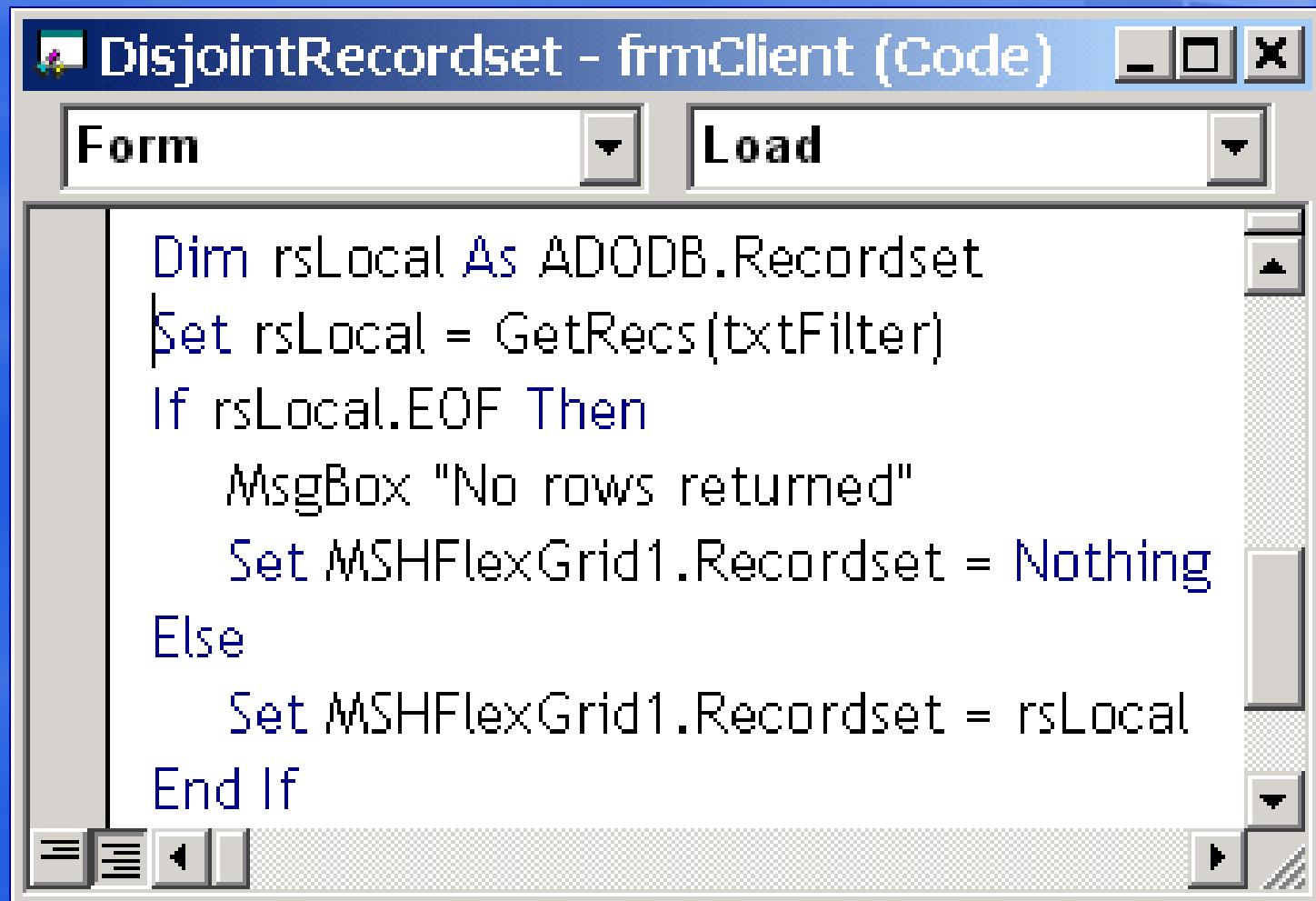
The screenshot shows a Visual Basic code editor window with the title bar 'DisjointRecordset - GetRecsClass (Code)'. The window has two tabs: '(General)' and 'GetRecs', with 'GetRecs' currently selected. The code editor contains the following VBA code:

```
Public Function GetRecs(Author As String) As Recordset
    Dim rs As Recordset
    OpenConnection
    Set rs = New Recordset
    With rs
        .CursorLocation = adUseClient
        .Open "select Au_id, Author, Year_Born " _
            & " from authors " _
            & " where author like " & Author & " ", cn, _
            Options:=adCmdText
        .ActiveConnection = Nothing
    End With
End Function
```

The code defines a public function 'GetRecs' that takes a string 'Author' as input and returns a 'Recordset'. It uses 'OpenConnection' to establish a connection, creates a new 'Recordset' object 'rs', and then uses a 'With' block to configure the recordset's cursor location, open the SQL query, and set the active connection to 'Nothing'.

Passing Recordsets...

Client-side code



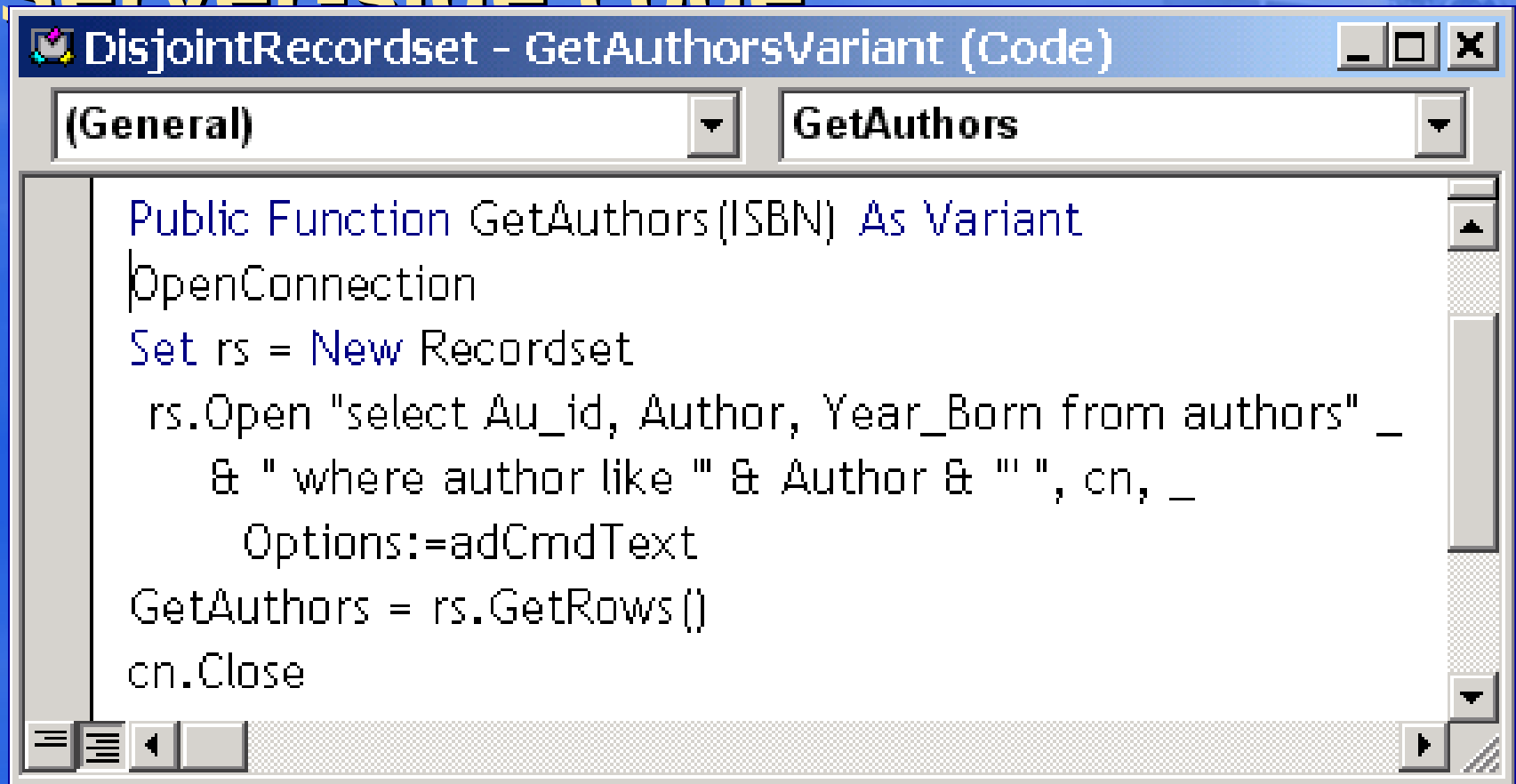
```
Dim rsLocal As ADODB.Recordset
Set rsLocal = GetRecs(txtFilter)
If rsLocal.EOF Then
    MsgBox "No rows returned"
    Set MSHFlexGrid1.Recordset = Nothing
Else
    Set MSHFlexGrid1.Recordset = rsLocal
End If
```

Passing Variant Arrays

- **Use GetRows to offload data**
- **Returns variant array**
 - **Uses Unicode - 16-bits/character**
 - **High row overhead**
 - **Partial DDL**
 - **No Field Names**
 - **No Row status**
 - **Packed with filler-characters**
 - **Check NetMon**
- **Use UBound to determine record count**

Passing Variant Arrays...

Server-side code

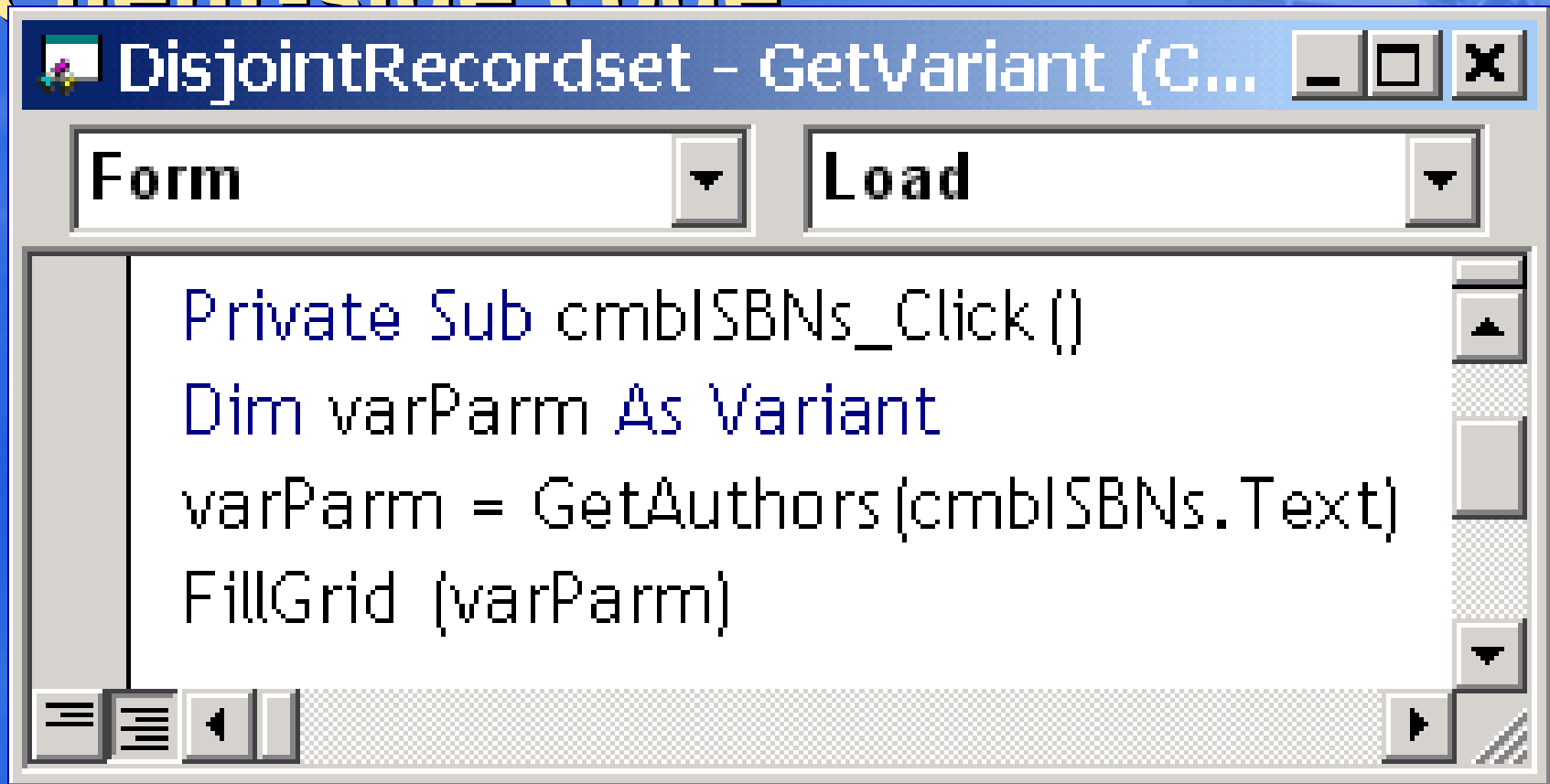


The screenshot shows a Visual Basic code editor window with the title bar 'DisjointRecordset - GetAuthorsVariant (Code)'. The window has two tabs: '(General)' and 'GetAuthors', with 'GetAuthors' being the active tab. The code in the 'GetAuthors' tab is as follows:

```
Public Function GetAuthors(ISBN) As Variant
OpenConnection
Set rs = New Recordset
rs.Open "select Au_id, Author, Year_Born from authors" _
    & " where author like '" & Author & "' ", cn, _
    Options:=adCmdText
GetAuthors = rs.GetRows()
cn.Close
```


Passing Variant Arrays...

Client-side code



Passing Delimited String

- Use GetString to offload data
- Pseudo delimited strings

Tab
Field
delimite

Using & Porting Gnu Cc, Version 2.3 1993 1-8821141-9-1 700 50

Success With Visual C++ 1995 1-8841330-9-6 95 44.95

CRLF
Row
delimite
r

- Low row overhead - No DDL
- Assumes you know what's coming
- Can be applied to clip property
- Record count - on your own...

User-Defined Data structures

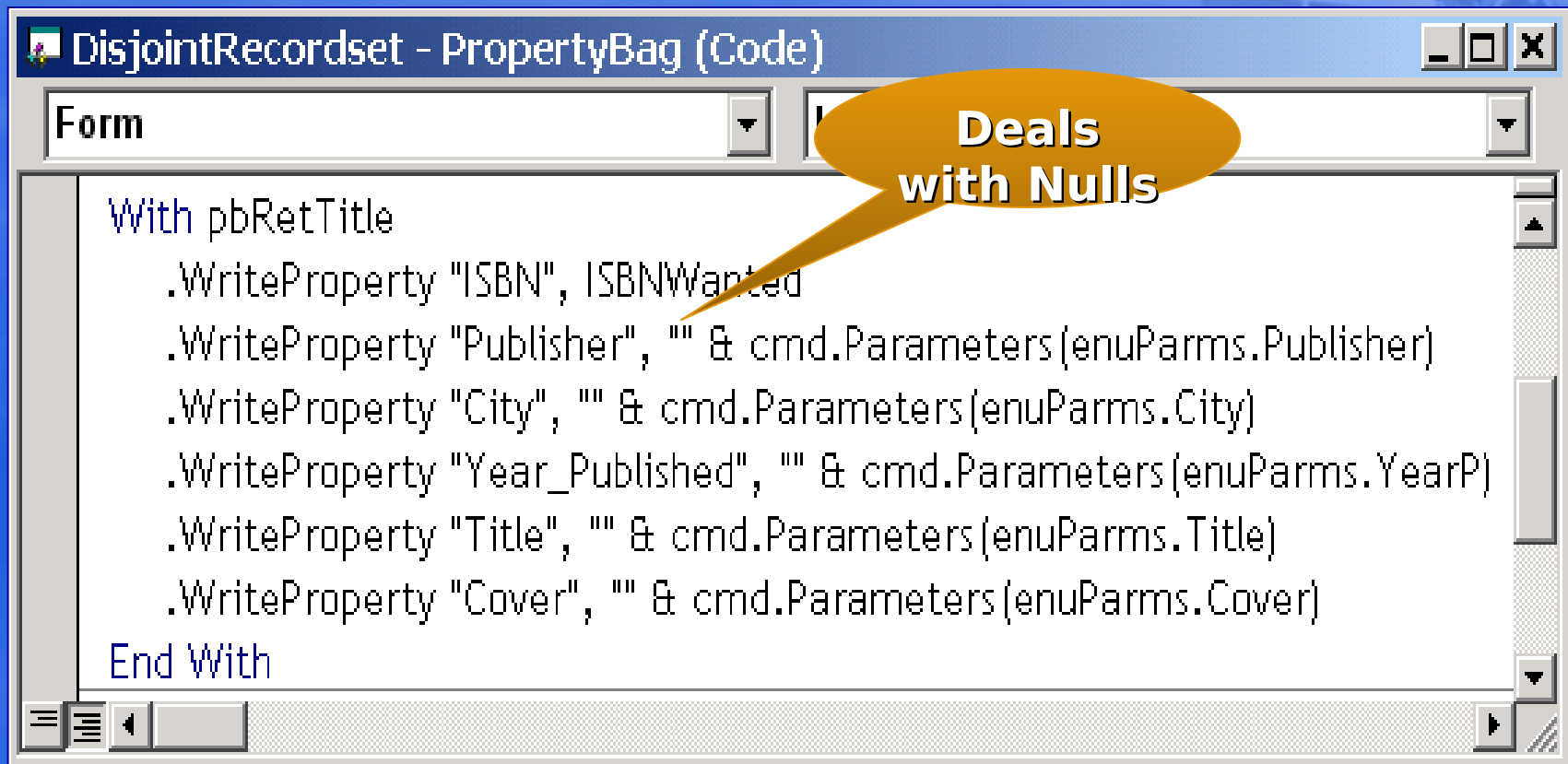
- **Pass**
 - **Single Type structure**
 - **Array of Type**
- **No ADO, COM, data control support**
- **Low row overhead - little DDL**
- **Record count - on your own...**
- **Assumes you know what's coming**

Passing PropertyBag Objects

- **Pass single Type structure or array of Type**
- **No ADO, control support**
- **COM structure referencing**
- **Low row overhead - little DDL**
- **Record count - on your own...**
- **Assumes you know what's coming**

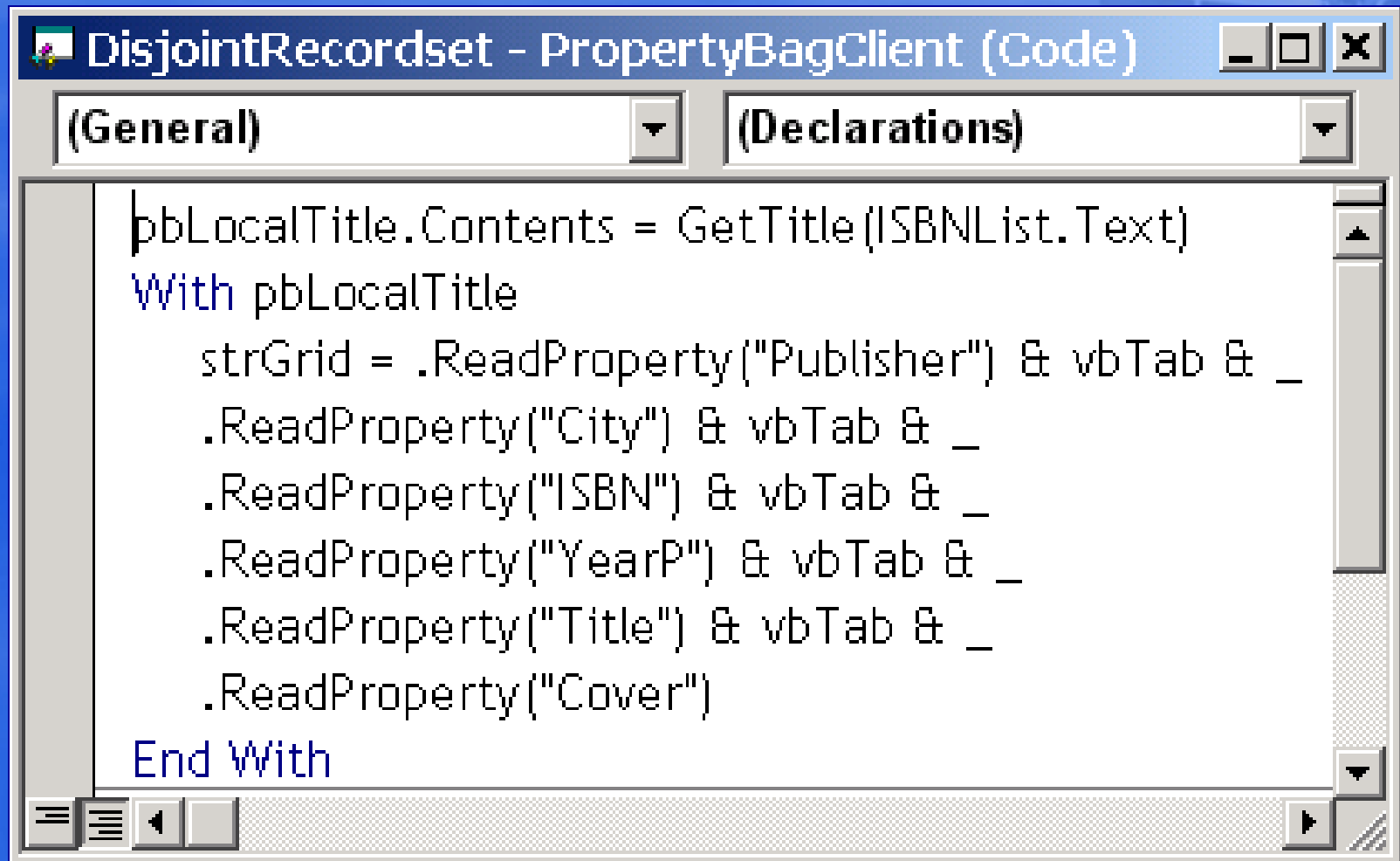
PropertyBag Objects

Server-side code



PropertyBag Objects

Client-side code



```
pbLocalTitle.Contents = GetTitle(ISBNList.Text)
With pbLocalTitle
    strGrid = .ReadProperty("Publisher") & vbCrLf & _
    .ReadProperty("City") & vbCrLf & _
    .ReadProperty("ISBN") & vbCrLf & _
    .ReadProperty("YearP") & vbCrLf & _
    .ReadProperty("Title") & vbCrLf & _
    .ReadProperty("Cover")
End With
```

Optimizing For Performance

- **Performance**
 - **What does it mean?**
 - **How is it measured?**
- **Achieving performance goals**
 - **Through smarter designs**
 - **Through smarter coding**
 - **Through smarter users**

Optimal Performance

- **Requires smarter**
 - **Designs**
 - **Implementations**
 - **Developers**
 - **Managers**
 - **World-class tools and engines**

POWER

UP



Microsoft®